SEM^2 Suite — Towards a Tool Suite for Supporting Knowledge Management in Situation Awareness Systems

Andrea Salfinger, Daniel Neidhart, Werner Retschitzegger, Wieland Schwinger
Department of Cooperative Information Systems
Johannes Kepler University Linz
Altenbergerstr. 69
4040 Linz, Austria
{andrea.salfinger, daniel.neidhart, werner.retschitzegger, wieland.schwinger}@cis.jku.at

Stefan Mitsch
Computer Science Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA, USA
smitsch@cs.cmu.edu

***Abstract*—Human operators in today's control centers, such as air or road traffic control, need to monitor a plethora of information obtained from diverse sources. To support them in detecting critical situations within this information flood and taking timely actions, operators thus need adequate information fusion and decision support systems. Research efforts on such dedicated Situation Awareness (SAW) systems have concentrated on assisting the operator in managing the current situations. However, little focus has been so far on integratively supporting the different phases of *knowledge management* in *SAW systems*, which encompasses the *acquisition*, *representation*, *validation*, *maintenance* and *reuse* of knowledge gathered for and during the use of these systems, such as configuring and maintaining suitable situation templates and exploiting already assessed situations. If operators and domain experts are not supported in these tasks, however, this may discourage them from a successful adoption of such systems in real-world control center applications, as user studies revealed. Based on these, and the lessons learned from the application of our SAW system implementations BeAware! and CSI to the domain of road traffic control, we therefore propose a first step towards a tool suite fostering knowledge management in SAW systems, which stretches from the configuration phase of the system to its runtime maintenance in the light of evolving environments and user needs.**

I. Introduction

Situation Awareness Systems. Human operators in today's control centers (e.g., air or road traffic control) need to monitor a plethora of information obtained from diverse sources. To assist them in detecting critical situations within this information flood and taking timely actions, they need suitable fusion of information and support for decision making, as provided by so-called Situation Awareness (SAW) systems. By reporting event combinations that require the operator's focus, i.e., *situations*, these systems reduce the cognitive load on the operator by targeting her attention towards such critical occurrences. *Template-based* SAW systems [1] allow for the specification and detection of specific *situation types* (STs) of interest and have proven their value for detecting a priori known, recurring patterns in a range of different application domains (e.g., maritime surveillance tasks [2] or road traffic monitoring [3]). In this sense, they are a kind of knowledge-based system, as these STs formulate the knowledge about the sought-after real-world situations and must be provided by domain experts. However, their usefulness hinges on the adequate specification of the STs of interest, i.e., the maintenance of that situational knowledge.

Knowledge Management. Consequently, a need for dedicated specification tools has been recognized from the earliest developments on template-based SAW systems on [4]. However, user studies as well as our experiences with our own SAW frameworks indicated that the application to real-world problem domains requires adequate support beyond mere representation of the situational knowledge, i.e., STs. Firstly, tool support for establishing of the *appropriate* situational knowledge base (KB) is needed. This is since: (i) Domain experts need to transform their potentially tacit knowledge to explicit representations of the situations of interest, but may be hampered by the *knowledge acquisition bottleneck*, as user studies on control center operators revealed [5]. Especially the specification of evolving situations, i.e., the behavior of a situation over time, has been identified to be a difficult task. (ii) Current tools pursue a purely top-down approach, thus lack support for determining the "fit" of the specified situational knowledge to real world data (i.e., *validating* the suitability of the represented situational knowledge), as well as *exploiting* the situational knowledge gathered during runtime, such as the encountered situation instances and tracked operator actions and preferences, which could be reused during situation assessment for refining predictions and action recommendations. Secondly, tool support for *maintaining* the situational KB is needed. This is due to the i) ever-evolving nature of both the environment under control, but also ii) changes of the system and iii) its usage by control center operators, therefore a one-shot *establishment of the situational knowledge* turns out to be insufficient, requiring a *continuous adaption and refinement of the situational KB*. However, this is scarcely supported in current SAW systems, as our survey on the evolution support of SAW systems revealed [6]. Overall, tool support for an integrative situational knowledge management for SAW systems involving the human domain expert in the loop is currently lacking.

Contribution. Therefore, in this paper we introduce our concepts on SEM^2Suite, a tool suite supporting the broader management of situational knowledge for SAW systems.

SEM^2Suite assists domain experts in (i) specifying STs by (ii) fostering knowledge acquisition and (iii) validation regarding potential situations of interest, (iv) maintaining the SAW system, i.e., assure its "fit" to evolving environments and usage needs, and information use and reuse, as it allows to (v) explore and (vi) systematically exploit encountered situations and undertaken actions.

Structure of the Paper. In the next section, we attempt at identifying the different tasks regarding the management of situational knowledge encountered in SAW systems, exemplified on our own previous work on a SAW framework for control center applications. In Sec. III, we discuss related work addressing knowledge management (KM) issues of SAW systems, before introducing SEM^2Suite, a tool suite for KM in SAW systems, in Sec. IV. Finally, Sec. V presents an outlook on future work.

II. Knowledge Management Tasks in SAW Systems

In literature, a series of different KM processes comprising partly different tasks can be found (e.g., [7], [8], [9], [10]) — however, w.r.t. managing the knowledge relevant for template-based *SAW systems*, up to now no common understanding has been gained so far. Although Jakobson's framework of Situation Management [11] serves as a valuable basis and addresses some KM-related issues of SAW systems (such as highlighting the central role of the *Situation Model*, and characterizing the concepts of *Situation Acquisition* and the *Situation Memory*), its aim is on characterizing the different functional processes and tasks necessary in SAW systems. No further characterization is provided w.r.t. the specific tasks required to build and maintain the situational KB.
Thus, basing on this broad body of literature, in the following we elaborate on those KM tasks, and their challenges and open issues, which we identified as foremost relevant. We base on findings reported in literature and our experiences on evaluating our own template-based SAW-Systems BeAware![1] [3], [12], [13] and CSI[2] [3], [14] in the domain of road traffic management (RTM), supported by our project partner, Austria's highway agency ASFINAG[3]. In these ontology-driven SAW systems, domain experts transfer their *domain knowledge* to the SAW system by specifying STs representing the sought-after real-world behavior (thus populate the *Model* KB, as shown in Fig. 1), which are then translated to rules. During *situation assessment* (SA), the SAW system's *Inference Engine* matches *data* obtained from the *observed environment* against this rule base. Matched rules trigger the creation of a *situation instance* of the corresponding ST, which is reported to the *control center operator*, and stored in the (Situation) *Memory* of the KB.

Knowledge Representation doubtlessly denotes one of the core knowledge-related tasks in template-based SAW systems. It encompasses the modeling of the domain knowledge, usually encoded by means of ontologies, and the specification of the STs of interest in terms of this domain ontology. Thus, this phase requires both a suitable knowledge representation

[1] situation-awareness.net
[2] csi.situation-awareness.net
[3] www.asfinag.at

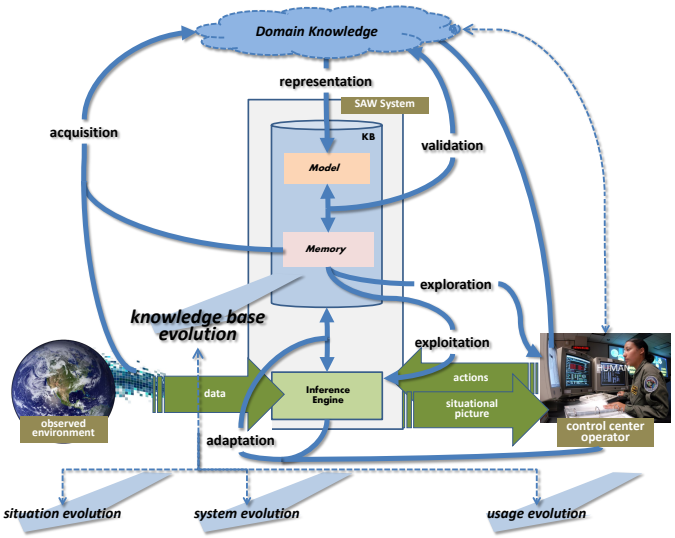

Figure 1: The different tasks for managing situational knowledge in SAW systems.

for situations, i.e., situation model for specifying STs, as well as dedicated tools supporting a user-friendly specification of these STs. Different tools have been proposed which aim at supporting domain experts in specifying STs (e.g., [2], [4], [15]). These mostly employ the JDL situation model [16] as knowledge representation model for STs, which, however, does not provide sufficient account of evolving situations, as has been acknowledged recently [5]. Only [2] supports the specification of *evolving* situations, however, does not allow to characterize the different evolutionary phases (e.g., *trigger*, *climax* and *clearance* phase, as distinguished in [12]), nor criticalities. Whereas highlighting critical, on-going situations supports human operators in mitigating them in a *reactive* way, the ultimate goal of a SAW system should be to provide *proactive* or *prescriptive* support, i.e., alert the operator already if a situation may be *developing towards a critical situation*, thereby, allowing the operator to prevent the situation from escalating. Therefore, BeAware! supports the specification of STs along their different evolutionary phases, as the domain expert could formulate dedicated *trigger*, *climax* and *clearance* STs. This, however, yields different situation instances for each phase. In the course of evaluating the results obtained on real-world data sets, we learned that we would need a *situation evolution model* (SEM) allowing to model situations across their *entire lifecycle* (as proposed in [17], [18]) to enable tracking and analysis of *evolving real-world situation instances*.

Knowledge Acquisition. The problem of *Knowledge Acquisition*, i.e., the elicitation of the domain experts' knowledge, has been widely acknowledged in the SAW system community [19], [20]. Prior to specifying STs, the question arises of *what* actually would be STs of relevance in the corresponding application domain. This is complicated by the fact that highly skilled, experienced experts face more difficulties in articulating their know-how than novices. Furthermore, the more tacit knowledge is, the more valuable it tends to be [21]. This is a crucial factor for template-based SAW systems, which depend

on the externalization of domain experts' and operators' know-how. User studies involving control center operators revealed that the problem is even more severe regarding the specification of *evolving situations* [5]. In order to support domain experts in leveraging their tacit knowledge, suitable means for knowledge discovery and data mining facilities should be provided [19], [20]. Although data mining techniques have already proven their capability of discovering important information for SAW applications, existing tools require expertise in data mining. Therefore, domain experts would need to be provided with preconfigured analysis functions [22].

From our experience with BeAware!, we can report that especially the aim to detect evolving situations from their *trigger* on introduces novel challenges: Whereas *climax* situation states are typically known, their preconditions are often not known in sufficient detail. This indicates the need for supporting an *incremental acquisition* of the required situational knowledge, starting from the specification and assessment of *climax* situation states and providing a "rewind" functionality allowing to step back in time to investigate on their preconditions.

Furthermore, in our evaluation domain of RTM it turned out to be completely unrealistic to assume that every potential situation of interest can be specified a priori. We need to expect to encounter novel, previously unknown situations at runtime. Therefore, analogous to the human operator, the SAW system must be capable of reacting to the unexpected. However, whereas (data-driven) anomaly detection based techniques for situation assessment have proven their superiority in these tasks (e.g., [1], [23]), and *hybrid* SAW systems have been proposed that include both, rule-based as well as anomaly detection based situation assessment modules [24], [25], current SAW systems do not provide means to transfer the results obtained with anomaly detection methods to rule-based situation assessors (as suggested for future work in [25]). This would allow to convert the unanticipated to novel knowledge, or to refine existing STs (e.g., if previously unknown preconditions or evolutions are detected), thereby maintaining and enhancing the situational KB. In the light that recent information fusion models emphasize the need for an integrated approach of combining data analysis with template-driven approaches [26], concepts for supporting the acquisition of novel knowledge based on data analysis techniques are needed.

Knowledge Validation. Whereas *knowledge representation* issues have been addressed by providing various ST specification tools, we have not encountered tool-supported means to *validate* the specified STs. However, our experience in the application of BeAware! to real-world control center settings indicated a need for considering these issues. Domain experts should be enabled of validating the modeled STs w.r.t. their *syntactical* and *semantic* correctness (i.e., the specification tool should provide means to check that the relations specified within a ST are non-contradictory). Furthermore, we observed that even a semantically correct ST is of limited use if it fails to capture the real-world behavior of interest, which we term the *adequacy* of a ST. For instance when formulating a ST "An accident causes a traffic jam", our system delivered doubtfully few hits — fewer than would be expected by a domain expert. As a matter of fact, this was due to improper specifications, as the mental model of the domain expert did not fully reflect the behavior observed in the data, requiring to adapt and assess the specifications several times. Therefore, we conclude that domain experts require means to assess whether their mental models "fit" the observed data, i.e., means to *validate* their a priori domain knowledge w.r.t. available domain data sets.

Knowledge Adaption. To retain their usefulness over prolonged application periods while interfacing dynamic environments, SAW systems also need to account for various kinds of evolution: Due to the underlying *environment's evolution*, the STs' specificity may change over time, i.e., once specified STs may fail to capture situations of relevance. Operators thus need to detect this *concept drift* [27] and adapt the STs accordingly. Therefore, a one-time configuration of STs would be insufficient, requiring a *configuration management of information models* [26], otherwise the system's usefulness decreases over time. Niklasson et al. even concluded that a fully automatic system would need to maintain a state similar to SAW on its own [28]. To remain of high value for the operators, the SAW system needs to maintain the maximum possible degree of "fitness", denoting the suitability between knowledge (i.e., the STs) and the problems confronted (detecting and preventing critical situations in a dynamically changing environment).

Furthermore, as template-based SAW systems require substantial configuration efforts, means must be provided to capture as much information as possible in a *non-intrusive* way. This could be realized by tracking the operator, thereby allowing the system to learn from the operators, as well as inferring the operators' preferences and changing usage (*usage evolution*) and adapting towards those. The specification and maintenance of STs could be simplified if the SAW system could autonomously, or in an operator-guided fashion, learn STs and actions of relevance from the operators. This is especially relevant as valuable tacit knowledge often results in some observable action, when individuals understand and subsequently make use of knowledge [21]. Thereby the tracking of these observable actions allows to dynamically acquire the experience and tacit knowledge of operators. Summarizing, in order to maintain the KB's "fitness" to the operators' tasks, *knowledge adaption* requires the system to address both *self-adaptivity* as well as the incorporation of *user-driven changes* at runtime (as also demanded in [26]).

Knowledge Exploration. Operators learn from previously encountered situations and thus gain more experience, which in turn enhances their future decisions. Thus, a SAW system should support *Investigative Situation Management* [11], i.e., provide the operators with means to explore and learn from the constantly growing *situation memory*. Operators and analysts should be enabled of exploring also distinct evolution patterns ("Why did a situation evolve in a specific way?", "What would be the typical evolution?"). Regarding our own experience, our domain experts also indicated the desire to assess the success of undertaken actions, which, in case of large-scale events, is conducted by means of after-action-reviews. Forensic analysis w.r.t. to undertaken actions (e.g., "What went wrong?", "What could have been done better?") thus requires a means to assess the effect of actions onto the evolution of encountered situations, thereby requiring that these actions are tracked and linked to the persisted situation evolutions.

Knowledge Exploitation. Furthermore, an intelligent SAW system should be capable of *Situation Learning* [11], i.e., *exploit* the Situation Memory to enhance its future SA and predictions. Currently, learning from previously encountered

situations is rarely addressed in rule-based SAW systems, as opposed to data-driven SA approaches, which by definition base on observed data, and are capable of learning if they are continuously incorporating new data (e.g., as in [29]). Moreover, to enhance the system's decision support capability (i.e., support *resolution* [30]) and aid operators w.r.t. to the question "What should I do?", the system should also be able to retrieve those actions that have yielded the most desirable outcome in the past. Thereby, it could capture and exploit the experience of operators by tracking their actions, similar to the approach of experience retrieval suggested in [31].

Summarizing, Fig. 1 outlines the interplay between the different KM tasks, which highlights the complexity of KM in SAW systems. KM in SAW systems is further complicated by the need for a rapid *KB evolution* due to *environment*, *usage*, and consequently *system evolution*. In the light of these challenges, operators and domain experts would benefit from tools supporting the identified KM tasks and addressing the discussed issues.

III. Related Work

In the present section, we elaborate on tools supporting KM issues in SAW systems.

Knowledge Acquisition. Whereas *knowledge acquisition* has been considered an important problem within SAW systems' research (cf. Sec. II), we did not encounter tools specifically addressing this issue for template-based SAW systems. The expert system and inference engine KASER [32] provides a step towards the generation of novel knowledge, by employing deductive and inductive reasoning to combine existing rules. Thereby, a continuously growing, virtual rule space is acquired, which is exponentially larger than the declared rule space. However, KASER requires domain experts to initially populate the rule base with relevant rules, which is supported by providing various input facilities and auto-completion features (basing on textual input), but not by means of data-driven suggestions.

Knowledge Representation. Several tools have been proposed that aim at facilitating the specification of STs for domain experts. These, however, provide still limited support regarding the specification of STs for evolving situations.
Matheus et al. introduced RuleVISor, a custom SWRL rule editor, which represents a central configuration component of their Situation Awareness Assistant SAWA [4]. RuleVISor allows to model STs in terms of a domain ontology, which are then translated to the corresponding rules.
Edlund et al. also developed a dedicated ST editor, a configuration component within their ontology-driven, rule-based SAW framework [2]. They especially highlight that it allows for the specification of *evolving* situations, by denoting whether the relations employed within a ST need to hold concurrently or subsequently. However, this only allows for the specification of a single evolution path, as alternatives cannot be specified. Furthermore, as a crucial drawback, their rule engine implementation detects such situations only *after* they completed their overall evolution.
Costa et al. proposed SML, a *Situation Modeling Language* allowing for a graphical composition of STs, which are then automatically translated to JBoss Drools[4] rules [15]. However, whereas they consider the possibility of referring to *current* and *past situations*, a more fine-grained distinction of situation evolution is not supported. Although their approach allows for specifying behavior over time, only a single course of events can be specified. Therefore, it is not possible to specify a *branching* ST, which might evolve either one way or the other.

Knowledge Validation. None of the discussed approaches that allow for the specification of STs provide specific tool-supported means to directly assess one's definitions on available data during specification, in order to examine the validity and specificity of the proposed STs.

Knowledge Adaption. Furthermore, we did not encounter specific tool-based support regarding the maintenance of the (template-based) SAW system, such as detecting *concept drift*: Only Edlund et al. state the need that the system must be capable of incorporating both, long-term rules, as well as rules that need to be added temporarily, thus are only valid for a certain time frame [2]. However, they leave the maintenance of the temporal validity to the operators, as they do not foresee any automated support to detect when certain rules become outdated over time, or their specificity changes.
As a first step towards bridging the gap between template-based and anomaly detection SA approaches, Rhodes et al. introduced SeeCoast, an SAW system for coastal surveillance [24]. SeeCoast incorporates both, a rule-based situation assessor, as well as data-driven anomaly detection modules. Thereby, this SAW system supports both the detection of known, recurring situation patterns, as well as reacting to novel situations. However, whereas their anomaly detection SA module continuously adjusts itself to environmental changes, by gradually being re-trained on new data, and even learns from operator feedback, they do not discuss means whether and how the *anomalous situations* could be further on integrated in the existing rule-base. Thus, they do not provide any suggestions to evolve the system's KB. A similar approach of combining rule-based SA with anomaly detection techniques has been described in [25], which have been neither integrated. This is, however, suggested as future work.

Knowledge Exploration. Approaches that allow to assess the situational knowledge the SAW system has gathered over time, or allow to explore what the system has learned, are rarely found. Riveiro et al. proposed a visualization interface that jointly visualizes normal behavioral statistical models (generated from the data) and expert rules (specified by domain experts) within a "scatter plot grid", thereby providing a means to explore the "knowledge space" of the system [25]. Their visualization aims at exhibiting how well the experts' knowledge correlates with the models extracted from the data.

Knowledge Exploitation. Currently available template-based SAW systems do not attempt at systematically reusing the situation instances accumulating in the situation memory to refine SA or projection. An interesting approach regarding the exploitation of experience from expert operators has been suggested in [31]: Operators' decisions, i.e., actions, are continuously tracked, along with the current context (employing a dedicated *Action-Observation-Hypothesis* model), and stored

[4] http://drools.jboss.org

in a KB. This aids junior operators who can retrieve these experiences for similar contexts.

Concluding, whereas interesting approaches have been proposed addressing specific KM tasks, we did not encounter approaches that comprehensively support different KM tasks in template-based SAW systems, i.e., stretch across *knowledge acquisition*, *knowledge representation*, *knowledge validation*, *knowledge adaption*, *knowledge exploration* and *knowledge exploitation*. Therefore, as a first step towards providing a KM tool that integratively supports distinct situational KM tasks, we elaborate on a tool suite that complements SAW systems by supporting dynamic situational KM.

IV. A Tool Suite for Knowledge Management in SAW Systems

In the present section, we propose SEM^2Suite, a modular (situation evolution modeling and maintenance) tool suite (cf. Fig. 2) complementing SAW systems, which supports KM tasks in rule-based SAW systems, thus contains modules for:
Knowledge Acquisition: comprises components suited to explore and mine existing domain data sets, in order to *acquire* insights about potentially relevant STs, as well as anomaly detection components that aim at detecting not yet specified situations, which may be transferred to the KB (cf. Fig. 2 Ⓐ).
Knowledge Representation: comprises a specification pane that allows to graphically compose *Situation Evolution Types* (SETs), based on assembling object types and relation types from a suitable domain ontology (cf. Fig. 2 Ⓑ, Fig. 3).
Knowledge Validation: comprises a set of different components for supporting the validation of the previously modeled SETs w.r.t. their *syntactical* and *semantic* correctness, as well as their *adequacy*, i.e., their "fit" to and specificity on real-world domain data sets (cf. Fig. 2 Ⓒ).
Knowledge Adaption: comprises components that assure self-adaptivity as well as the incorporation of user-driven changes at runtime, such as components that detect *concept drift* within the monitored environment influencing the specificity of the modeled SETs, components that allow operators to activate, deactivate or modify specific SETs, performance monitoring components that seek to optimize the situation assessor's performance, and user tracking components that aim at optimizing the UI towards the user's preferences (cf. Fig. 2 Ⓓ).
Knowledge Exploration: comprises (visual) analytics interfaces that allow operators to explore the accumulated situation memory, i.e., allows to address questions such as "How do situations typically evolve?" or "What have been good actions in specific situations?", thereby also supporting forensic analysis (cf. Fig. 2 Ⓔ, Fig. 5).
Knowledge Exploitation: comprises components (cf. Fig. 2) that aim at reusing the experience accumulated in the situation memory by refining the prediction on currently observed situations and the recommendation of the most promising actions, based on comparing the current situations to similar situations observed in the past, which are retrieved from the situation memory. Thereby, it specifically supports real-time decisions (cf. Fig. 2 Ⓕ).

We realize this range of functionality within a flexible, configurable tool suite, as not every monitoring task may require each aspect, or may provide all prerequisites. For instance the components for *knowledge acquisition*, as well as *knowledge validation* w.r.t. *adequacy*, depend on the availability of a domain data set. If no data set is available, SETs can only be modeled in a top-down fashion, and be fitted towards the data at runtime. Finally, it should be emphasized that SEM^2Suite abstracts the modeling concepts from concrete technologies and implementation details, thereby allowing to interchange these technologies. In the following, the different components will be explained in more detail.

A. Knowledge Acquisition

SEM^2Suite's ***Data Analyzer*** component represents a container for interfacing various established data mining tools suitable for data from SAW systems, such as R[5] and Weka-STPM [33], however, provides a dedicated user interface tailored towards SAW data sets and applicable, preconfigured analyses. The employed knowledge discovery tools support human operators in gaining ideas about potential STs, by detecting aberrations from the normal environmental picture, which often correspond to situations of relevance [1], rare and unusual events, and often co-occurring object types. Furthermore, STs may be autonomously suggested by means of association rule mining [34], which therefore retrieves objects in specific relations that may serve as initial ST suggestion that can be refined by the domain expert.
The ***Data Analyzer*** further comprises anomaly detection modules that report unusual courses of events at runtime, which may correspond to novel situations. If the operator decides that a reported anomaly corresponds to a novel SET, the ***Data Analyzer*** extracts the object types and relations between the encompasses objects, and provides the operator with an initial SET suggestion, that can be confirmed or refined by the operator, before it is incorporated into the SET KB. Thus, the human operator's expertise can be continuously incorporated.

The ***Hotspot Analyzer*** supports the incremental specification of SETs: Operators may begin with specifying the *climax*, i.e., *hotspot*, situation state, and let the system assess the hotspot situation states on the supplied domain data set. Based on the found situations, the system may automatically try to infer typical preconditions of these (how were the objects being part of the hotspot related in earlier timesteps?), and may already provide the operators with a template including the inferred, potential relations and object types. On the other hand, if automatic SET suggestion fails, the operator should be enabled to sift through the provided examples (by providing a "rewind" functionality to assess the situation's history) and use her experience to infer the likely causalities.

B. Knowledge Representation

A suitable knowledge representation lies at the very heart of Situation Management (SM) [11], i.e., a model for specifying templates for real-world situations of interest, and allows the tracking and persisting of these situations in order to enable both investigative as well as predictive Situation Management. To address the *knowledge representation* challenges regarding evolving situations reported in Sec. II and account for the dynamic evolution of real-world situations, we therefore proposed a dedicated Situation Evolution Model (SEM) in our previous work [18], which constitutes a knowledge representation that

[5] http://www.r-project.org

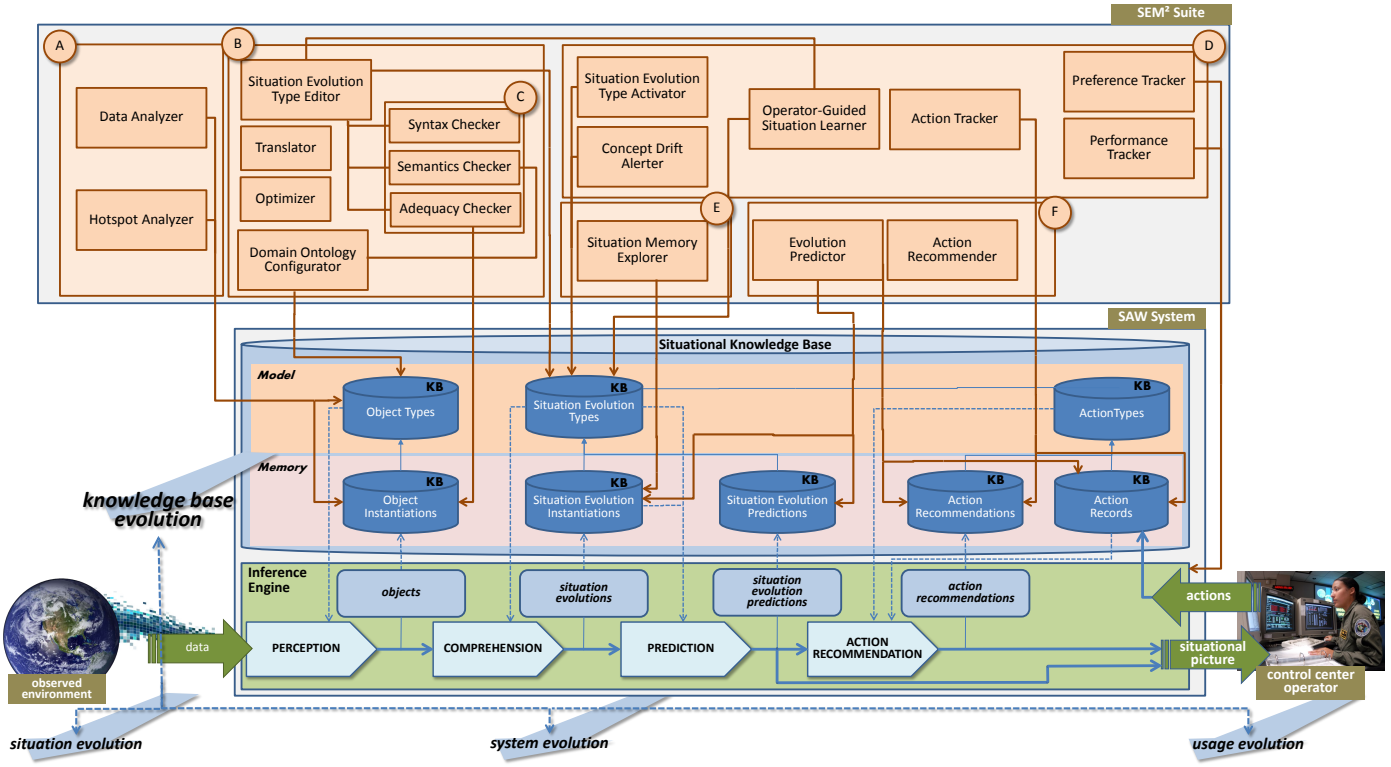

Figure 2: An architectural overview on SEM^2Suite's different components and how these interface and interact with a SAW system's *Situational KB*.

allows to model, track and reason upon evolving situations. Therefore, situation evolution types (SETs) specified by instantiating the SEM for a specific evolving ST of interest denote the basic *knowledge representation* utilized by SEM^2Suite.

Domain Ontology Configurator. The specification of SETs requires a suitable domain ontology. In order to allow for a domain-agnostic SA and SEM^2Suite functionality, we therefore require to engineer this domain ontology by extending a generic, i.e., domain independent *SAW core ontology* (based on [35]). This SAW core ontology characterizes the basic entities of relevance in SAW applications, notably *Object Types* sensed from the environment, which are characterized by spatial and temporal properties (e.g., a traffic jam which possesses a specific location and duration), *Relation Types* indicating semantic relationships between such *Objects*, such as spatio-temporal relations (e.g., basing on well-known relation calculi such as Allen's temporal relations [36] or the Region Connection Calculus [37]), and the SEM, which allows to model evolving situations. Thus, to develop a concrete SAW control center application based upon this framework, the SAW core ontology needs to be extended by a suitable *domain-specific ontology*, in order to allow for sensing and reasoning about the information of relevance of the corresponding domain. Furthermore, the basic spatio-temporal relations provided by the framework need to be configured by *parametrizations* suitable for the given application domain. For instance, suitable threshold intervals for spatial relations such as *Close* and *Far* need to be determined, which may substantially differ between a maritime traffic control center and a highway agency.
In our current prototypical implementation, we follow a UML-based approach of ontology-engineering [38], by employing a dedicated UML modeling tool[6] for generating a Java class library comprising the object ontology, database schemes and corresponding Hibernate[7] mappings. Ontology engineering is thus currently decoupled from SEM^2Suite, which expects to be provided with a Java Archive (jar) comprising the data access layer generated by the employed modeling tool.

SET Editor. The SET Editor allows for the graphical specification of Situation Evolution Types (SETs). Fig. 3 shows a screenshot of our current prototypical implementation of the ***SET Editor***, which features the diagram for the specified SET "Wrong-way driver approaching tunnel" (WDApproachesTunnel). A SET captures the potential evolutions of real-world situations by means of a state-transition system, whereby the states correspond to a particular *relational* state within the evolving situation, for instance the state of affairs where an object of object type "Wrong-way driver" (WD) is in a relation type "Close" to another object type "Tunnel" (depicted by the correspondingly named circles in the diagram shown in Fig. 3). As a monitored real-world situation instance evolves, in subsequent time steps, at one point the relation between the observed objects may change, for instance the monitored wrong-way driver object may come "very close" to the tunnel, which is modeled as a *transition* from the situation state type (SST) "Wrong-way driver close tunnel" to the SST "Wrong-way driver very close to tunnel". In order to reference a specific object across different SSTs, i.e., refer to it throughout its

[6]Visual Paradigm for UML, http://www.visual-paradigm.com
[7]http://hibernate.org

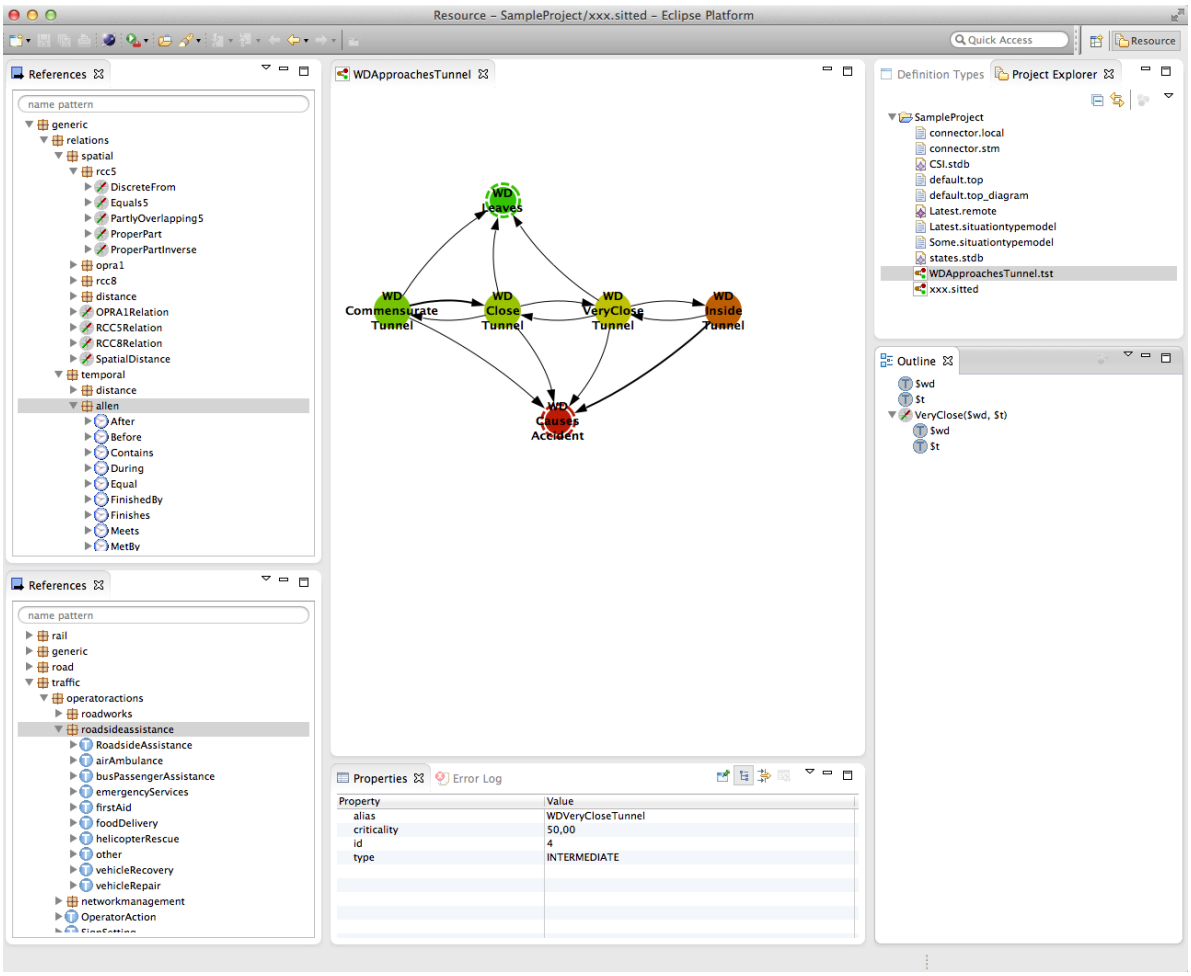

Figure 3: A screenshot of the current SEM^2Suite prototype, which shows the ***SET Editor***.

evolution, the different object types must be identifiable by means of an alias (which is unique within a specific SET), as shown in the Outline View in the lower right section of the editor screenshot, which highlights the different object references (i.e., object types identified by an alias) and relations employed within a specific SET. A domain expert thus models the different relational states of evolving situations, i.e., SSTs, by assembling them from object types and relation types from the employed domain ontology, which is comprised in the two views on the left side (the upper one comprises distinct relation types, the lower one comprises the object ontology), and specifying the evolution transitions between these SSTs afterwards. Real-world situations tracked at runtime correspond to a specific *path* throughout their SET, i.e., are persisted as a list of interlinked SST instances created by matched SSTs (along with the temporal information how long each of these "situation snapshots" lasted). Furthermore, the domain expert can specify additional properties for those SSTs, such as their evolutionary phase and criticality (visualized by means of the different colors of the states shown in Fig. 3). SEM^2Suite fosters the reusability of modeled components by allowing to include already specified SSTs in other SETs. Furthermore, different action types can be specified for a SST, such as "closing the tunnel for incoming traffic" (and encompassed actions, such as "setting the traffic lights") and "informing emergency units", which can be suggested to the operator as soon as a situation instance of that specific SST is detected.

Translator. Finally, each modeled state needs to be translated to a rule according to the language the SAW system's rule engine employs, such as JBoss Drools Expert[8] or Jess[9]. Therefore, SEM^2Suite requires a dedicated ***Translator*** library that allows to attach the translation rules how a SST pattern can be translated into the target rule language, i.e., a IF-THEN pattern. The left-hand side of the rules (IF) needs to match the Object Types i.e., check the type of the matched object and bind it to the given *alias*. Furthermore, a specific translation for each Relation Type must be implemented.

Optimizer. Since a SST comprises a *set* of relations, no ordering is implied on these relations. Regarding the evaluation of IF-THEN rules, however, the different IF-clauses are processed in sequential order. Thus, the ordering of the IF-clauses matters in the respect that a clause that significantly reduces the number of matched objects should be prioritized, as it raises the performance of the rule evaluation, as we evaluated in [3]. Therefore, a dedicated ***Optimizer*** strategy library encodes

[8]http://www.jboss.org/drools/drools-expert
[9]http://www.jessrules.com

different optimization patterns targeted at specific rule engines and application domains. These may provide rules on the sequential ordering of relation evaluation, such as spatial relations should be prioritized over temporal relations.

C. Knowledge Validation

Assuring *syntactical correctness* means to check that the specification complies with the chosen specification formalism, i.e., the SEM. The ***Syntax Checker*** may validate that Object Types are referenced by an alias, the arity of relations is met, and Object References overlap across a single evolution step, as demanded by the formalism described in [18].

The ***Semantics Checker*** goes one step further by assuring that the specifications correspond to meaningful and possible real-world situations, by taking into account the semantic interpretations of the relations. For instance a user may erroneously specify that an accident has occurred *inside* a tunnel and is located *in front of a tunnel*, which is contradictory. To enable SEM^2Suite to perform such semantic checks, this information must be provided in the domain ontology, by means of Conceptual Neighborhood Graphs (CNGs) representing the *epistemic knowledge* on how relations of different relation families can be combined (as elaborated on in our previous work on exploiting CNGs for situation projection [13]).

Further validation w.r.t. the *adequacy* of the specified SETs, i.e., determining whether the specified SETs are indeed capable of capturing the sought-after real-world situations, can be conducted with the ***Adequacy Checker***. This component needs to be provided with a database on data collected from the corresponding application domain, which is accessible from the data access layer loaded in the ***Domain Ontology Configurator***. The ***Adequacy Checker*** can be used to assess *Relation* parametrizations, such as providing histograms for the parametrizations of relation families, and object types of interest. For examples, one may be interested in how the specificity of the SET "Probably fusing traffic jams" defined by the relation "traffic jam j1 *close* to traffic jam j2" changes if the relation type is changes from *close* to *very close*, or if the threshold intervals of these relation types are adapted (e.g., if the threshold interval for the relation type *very close* is changed from 0 - 1 km to 0 - 2 km, thereby widening the set of matched relations). Thus, the ***Adequacy Checker*** pane provides *before* - *after* histograms, showing the number of situations assessed with the previous and the new definitions. Thereby, it provides a means to inspect and adjust the specificity of the specified SETs, as the domain expert is enabled to incrementally adapt the relation parametrizations and SET specifications, until the desired specificity is reached, depending on whether she wants to receive a multitude of situations, or a smaller set of results. As investigated on in [24], individual operators exhibit different preferences regarding the alarm rate, i.e., situation hit rate. Thus, SEM^2Suite thereby provides operators with a means to configure the SETs' specificity to their personal preferences.

D. Knowledge Adaption

The ***SET Activator*** provides an interface for operators to manage the activation and deactivation of SETs at runtime, thereby allowing the operator to enable and disable the assessment of specific SETs. New SETs can be added and existing

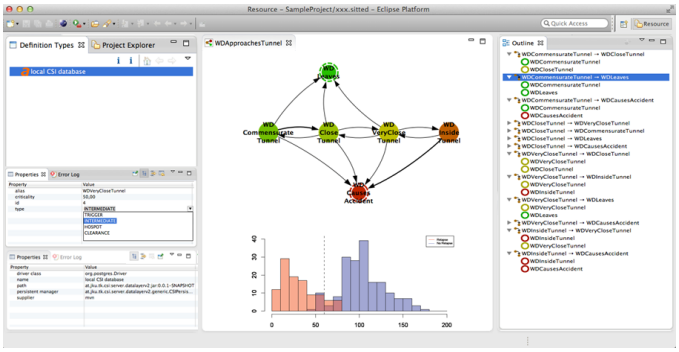

Figure 4: A tentative design draft of the ***Adequacy Checker***, which highlights the quantity structure observed from the data for different definitions (e.g., the quantity structure *before* and *after* changing the thresholds).

ones refined, as this component allows to launch the ***SET Editor*** upon a specific SET. However, such adaptions on-the-fly, during runtime of the SAW system, require the SAW system to keep track of the specific version of a SET a situation has been assessed with, therefore, requiring a SET *versioning* system, as well as *provenance tracking*.

The ***Concept Drift Alerter*** aims at preserving the desired specificity of the SETs, in spite of environmental evolution which may induce concept drift. Therefore, it continuously assesses the specificity of the activated SETs, and compares it to their common distributions stored in a KB. Furthermore, this component also stores the distributions of the environmental normalcy models, in order to detect changes within the normal environmental picture indicating concept drift. The ***Concept Drift Alerter*** thus detects if the specificity of a specific SET changes, and reports it to the operator, who is provided with a summary of the encountered changes and thus can investigate on these changes and adapt the SETs accordingly.

The ***Performance Tracker*** aims at providing a continuous runtime tuning of the SAW system, i.e., addresses *system evolution*. By performing a fine-grained tracking of the quantity structure of matched objects and relations, depending on temporal and load contexts, this provides the basis for a detailed analysis on the optimality of the translated rules. For instance, it may have been assumed during configuration of the system that assessing the temporal relations before assessing the spatial relations would be faster than vice versa. However, the logs provided by the optimization tracker may reveal that assessing the spatial relations first would more significantly reduce the number of matches for the next clauses. Thus, the ***Optimizer***'s strategies can be adapted accordingly and existing SETs may be recompiled thereafter.

The ***Operator-guided Situation Learner*** analyzes the human operators' reporting procedures, such as the semantic grouping of objects forming a situation for reporting purposes (as performed by our RTM demonstrators), derives object types and computes their interrelations, and provides the domain expert with these SET suggestions, who can again refine these suggestions.

The ***Action Tracker*** records the actions undertaken by the operators, both routine monitoring actions, as well as actions

that represent a dedicated response to a specific situation, which are directly linked to the corresponding situation. Over time, the operator's experience can be thus collected by the system, and allows to asses which actions have proven to be beneficial in which situations.

The ***Preference Tracker*** logs the operator's interaction with the UI, in order to infer the operator's preferences and adjust the display of the operational picture accordingly (e. g., by only showing the operator's preferred layers and level of detail).

E. Knowledge Exploration

The ***Situation Memory Explorer*** provides *Visual Analytics* support for interactively exploring the *Situation Memory*, in order to assess the evolution patterns of actually observed situations. The user can define a context of interest by moving a spatio-temporal selection frame and different filters. The ***Situation Memory Explorer*** then constructs the corresponding query and retrieves situations matching the context of interest from the memory. These situations are mapped onto the graphical representation of their SET, by aggregating their different, linked situation snapshots onto their corresponding SSTs. This allows to highlight the common *evolution patterns* of the current context of interest, i.e., which developments are more likely than others, how long situations last on average etc. By modifying the context, for instance by panning and zooming the spatial selection frame (cf. Fig. 5), the user can assess how these evolution patterns may change under different contexts. The ***Situation Memory Explorer*** further allows to save a snapshot of the current context and result. The results obtained with different assessed contexts can be arranged in a scatter-plot like fashion, in order to compare and contrast the results. For instance, it may become apparent that situations in tunnel X tend to develop less critical than situations in tunnel Y and take less time to be resolved. Based on such insights, for instance, the causalities for these distinct evolution patterns may be investigated on, thereby helping to elaborate on countermeasures.

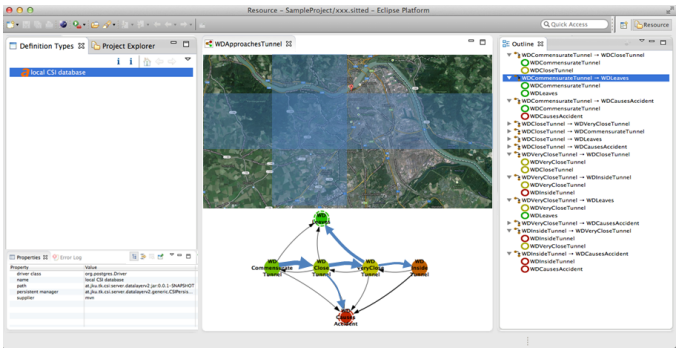

Figure 5: A design sketch of the ***Situation Memory Explorer***, showing the spatio-temporal selection pane which allows to highlight regions and periods of interest. The *Evolution Analysis Pane* (at the bottom of the main pane) highlights the evolution transition frequencies w.r.t. to the selection, as assessed from the *situation memory*, thus allowing the operator to investigate on and contrast local characteristics.

F. Knowledge Exploitation

The ***Evolution Predictor*** refines the projection of the currently observed situation based on retrieving similar situations, i.e., situations of the same SET, observed in the past. Since a SET models a state-transition system and all situations of a specific SET correspond to a path through this state-transition system, mapping the set of historic, similar situations onto the SET allows to derive the transition probabilities between its SSTs: Thus, the transition probability of the monitored situation's current SST to the SSTs reachable from this SST (i.e., potentially succeeding SSTs) can be computed. This can serve as a prediction estimate on the situation's most likely future development. For instance, the majority of historic "Wrong-way driver approaching tunnel" situations that evolved through SST "Wrong-way driver close tunnel" may have developed to SST "Wrong-way driver very close to tunnel", which can be visualized by a thicker linewidth of the SET's transitions (cf. Fig. 5), whereas only a small portion developed to SST "Wrong-way driver leaves the motorway". Thereby, when monitoring a "Wrong-way driver approaching tunnel" situation that is currently in the SST "Wrong-way driver close tunnel", it can be concluded that its evolution to the state "Wrong-way driver very close to tunnel" can be considered to be more probable, i.e., it is more likely that the situation escalates than that is resolves. The ***Evolution Predictor*** further could employ different *situation similarity measures*, which determine the set of historic situations that is considered for these computations: A *similar* situation may, in the most general case, be defined as a situation of the same SET (i.e., all historic situations of the same SET would be used, no matter how old), or, more specifically, as a situation of the same SET and also a *similar spatio-temporal context* (e.g., by only considering situations that occurred on the same weekday, the same time of day, and/or the same region). Furthermore, a weighting strategy could be employed, which specifies the influence of each historic situation instance within the prediction value computation. This allows, for instance, to strengthen the influence of recently observed situations over situations in the further past, thereby taking environmental evolution into account (i.e., recent situations are given a higher weight than older ones). Since it may be unclear which similarity measure and projection strategy provides the most accurate results, the ***Evolution Predictor*** thus needs to track its predictions (stored in the *Situation Evolution Prediction KB* shown in Fig. 2) and later on compare it with the actual developments, in order to be able to optimize its prediction strategy.

The ***Action Recommender*** realizes a kind of case-based reasoning functionality, in order to suggest the best action for a current situation. It retrieves situations similar to the current situation from the situation memory, analyzes which of the actions performed on those (retrieved from the *Action Records KB*) have led to the most favorable situation evolution, and recommends these to the operator. The operator is also enabled to inspect these historic situations, if she requires further details on how the past situations evolved. Action recommendations are stored in a dedicated KB (cf. Fig. 2), which allows to optimize action recommendations w.r.t. the operator's preferences (by comparing action recommendations with actually performed actions).

V. Conclusion and Future Work

In the present paper, we characterized the challenges of KM in SAW systems, based on findings in literature and experiences gained from the application of our SAW systems BeAware! and CSI to the domain of RTM. Based on the identified requirements, we sketched our vision on a tool suite fostering KM in SAW systems, which especially focuses on incorporating the dynamic aspects of evolving environments and user needs. Regarding future work, we aim at completing our prototypical implementation of this tool suite, and plan to evaluate the feasibility of our approach in a real-world case study involving control center operators from our demonstrators in the domain of RTM.

VI. Acknowledgments

This work has been funded by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under grant FFG FIT-IT 829598, FFG BRIDGE 838526, FFG BRIDGE 832160, WTZ/ÖAD AR18-2013 and UA07-2013.

References

[1] R. Laxhammar, "Anomaly detection for sea surveillance," in *11th Int. Conf. on Information Fusion*, 2008.

[2] J. Edlund *et al.*, "Rule-based situation assessment for sea surveillance," *Proc. SPIE*, vol. 6242, 2006.

[3] N. Baumgartner, S. Mitsch, A. Müller, W. Retschitzegger, A. Salfinger, and W. Schwinger, "A tour of BeAware – A situation awareness framework for control centers," *Information Fusion*, vol. 20, no. 0, pp. 155–173, 2014.

[4] C. Matheus *et al.*, "SAWA: An Assistant for Higher-Level Fusion and Situation Awareness," in *Proc. of SPIE Conf. on Multisensor, Multisource Information Fusion*, 2005, pp. 75–85.

[5] M. Nilsson *et al.*, "Extracting rules from expert operators to support situation awareness in maritime surveillance," in *11th Int. Conf. on Information Fusion*, 2008.

[6] A. Salfinger, W. Retschitzegger, and W. Schwinger, "Maintaining Situation Awareness Over Time - A Survey on the Evolution Support of Situation Awareness Systems," in *Procs. of the 2013 Conf. on Technologies and Applications of Artificial Intelligence (TAAI 2013)*. IEEE Computer Society, 2013, pp. 274–281.

[7] W. R. Bukowitz and R. L. Williams, "The knowledge management fieldbook," *Business Digest*, 1999.

[8] M. W. McElroy, *The New Knowledge Management: Complexity, Learning, and Sustainable Innovation*. KMCI Press, 2003.

[9] M. H. Meyer and M. H. Zack, "The design and development of information products," *Sloan Management Review*, vol. 37, pp. 43–59, 1996.

[10] K. Wiig, *Knowledge management foundations: thinking about thinking. How people and organizations create, represent, and use knowledge*. Arlington, Texas: Schema Press, 1993.

[11] G. Jakobson *et al.*, "A Framework of Cognitive Situation Modeling and Recognition," in *IEEE Military Communications Conf. (MILCOM)*, 2006.

[12] N. Baumgartner, W. Retschitzegger, W. Schwinger, G. Kotsis, and C. Schwietering, "Of Situations and Their Neighbors—Evolution and Similarity in Ontology-Based Approaches to Situation Awareness," in *Proc. of the 6th Int. and Interdisciplinary Conf. on Modeling and Using Context (CONTEXT)*. Springer, 2007, pp. 29–42.

[13] N. Baumgartner, W. Gottesheim, S. Mitsch, W. Retschitzegger, and W. Schwinger, "Situation Prediction Nets," in *Conceptual Modeling – ER 2010*, ser. LNCS. Springer, 2010, vol. 6412, pp. 202–218.

[14] N. Baumgartner, S. Mitsch, A. Müller, W. Retschitzegger, A. Salfinger, and W. Schwinger, "The Situation Radar: Visualizing Collaborative Situation Awareness in Traffic Control Systems," in *Procs. of the 19th World Congress on Intelligent Transport Systems*, 2012.

[15] P. D. Costa *et al.*, "A Model-Driven Approach to Situations: Situation Modeling and Rule-Based Situation Detection," in *2012 IEEE 16th Int. Enterprise Distributed Object Computing Conference (EDOC)*, 2012.

[16] J. Llinas *et al.*, "Revisiting the JDL Data Fusion Model II," in *Procs. of the Seventh Int. Conf. on Information Fusion*, 2004.

[17] M. Kokar *et al.*, "Situation tracking: The Concept and a Scenario," in *IEEE Military Communications Conf. (MILCOM)*, 2008.

[18] A. Salfinger, W. Retschitzegger, and W. Schwinger, "Staying Aware in an Evolving World — Specifying and Tracking Evolving Situations," in *2014 IEEE Int. Inter-Disciplinary Conf. on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. IEEE, 2014, pp. 172–178.

[19] J. Salerno *et al.*, "Building a framework for situation awareness," in *Procs. of the Seventh Int. Conf. on Information Fusion*, 2004.

[20] S. Das, "Addressing the knowledge acquisition issue for model based level 2/3 fusion," in *10th Int. Conf. on Information Fusion*, 2007.

[21] K. Dalkir, *Knowledge Management in Theory and Practice*. Taylor & Francis, 2013.

[22] M. Hadzagic *et al.*, "Maritime traffic data mining using R," in *16th Int. Conf. on Information Fusion (FUSION)*, 2013, pp. 2041–2048.

[23] B. Ristic *et al.*, "Statistical analysis of motion patterns in AIS Data: Anomaly detection and motion prediction," in *11th Int. Conf. on Information Fusion*, 2008.

[24] B. J. Rhodes *et al.*, "SeeCoast: persistent surveillance and automated scene understanding for ports and coastal areas," *Proc. SPIE*, vol. 6578, pp. 65 781M–65 781M–12, 2007.

[25] M. Riveiro and G. Falkman, "Interactive Visualization of Normal Behavioral Models and Expert Rules for Maritime Anomaly Detection," in *Sixth Int. Conf. on Computer Graphics, Imaging and Visualization CGIV '09.*, 2009, pp. 459–466.

[26] E. Blasch *et al.*, "Revisiting the JDL model for information exploitation," in *16th Int. Conf. on Information Fusion (FUSION)*, 2013.

[27] G. Widmer and M. Kubat, "Learning in the Presence of Concept Drift and Hidden Contexts," *Machine Learning*, vol. 23, no. 1, 1996.

[28] L. Niklasson *et al.*, "Extending the scope of situation analysis," in *11th Int. Conf.on Information Fusion*, 2008.

[29] B. Rhodes *et al.*, "Maritime situation monitoring and awareness using learning mechanisms," in *IEEE Military Communications Conf. (MILCOM)*, 2005, pp. 646–652.

[30] B. Mcguinness and L. Foy, "A subjective measure of SA: The Crew Awareness Rating Scale (CARS)," in *Proc. of Human Performance, Situation Awareness and Automation: User-Centered Design for the New Millenium*, 2000.

[31] C. Zhong *et al.*, "RankAOH: Context-driven Similarity-based Retrieval of Experiences in Cyber Analysis," in *2014 IEEE Int. Inter-Disciplinary Conf. on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*. IEEE, 2014, pp. 207–213.

[32] S. H. Rubin *et al.*, "KASER: a qualitatively fuzzy object-oriented inference engine," in *Procs. of the 2002 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, 2002, pp. 354–359.

[33] V. Bogorny *et al.*, "Weka-STPM: a Software Architecture and Prototype for Semantic Trajectory Data Mining and Visualization," *Transactions in GIS*, vol. 15, no. 2, pp. 227–248, 2011.

[34] ——, "Reducing uninteresting spatial association rules in geographic databases using background knowledge: a summary of results," *Int. Journal of Geographical Information Science*, vol. 22, no. 4, pp. 361–386, 2008.

[35] C. Matheus, M. Kokar, and K. Baclawski, "A core ontology for situation awareness," in *Procs. of the Sixth Int. Conf. of Information Fusion*, vol. 1, 2003, pp. 545–552.

[36] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, pp. 832–843, 1983.

[37] D. A. Randell, Z. Cui, and A. G. Cohn, "A Spatial Logic based on Regions and Connection," in *Computer*, 1992, vol. 92, pp. 165–176.

[38] P. Kogut *et al.*, "UML for Ontology Development," *Knowl. Eng. Rev.*, vol. 17, no. 1, pp. 61–64, 2002.